REMARKS

35 U.S.C. § 102 Claim Rejections

Challenger

By the Office Action dated August 23, 2006, the Examiner has rejected claims 1-27 under 35 U.S.C. § 102(b) as being anticipated by Challenger (U.S. Patent No. 6,026,413) (hereinafter "Challenger"). In order to be an anticipation of a claim under 35 U.S.C. § 102(b), a reference must teach every element of the claim, including the relationships between the elements. If any element is not fully taught by the reference, the rejection cannot be sustained.

Evaluating Challenger in this light, it is appropriate to examine the portions of Challenger which the Examiner has pointed to as teaching the claimed elements.

Claims 1-15

Claim 1

The Examiner has asserted that

> Challenger discloses in figures 1, 9, 12, and 30, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of transformation rules; a cache for temporarily storing transformed data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (Abstract; col.4, lines 6-10; col. 10, lines 14-24; col. 29, lines 32 +).

(See Office Action, page 2.)

To the extent the Examiner's language at page 2 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Challenger and claim 1:

Claim 1	**Challenger**
1. A client-server system capable of validating cached *eXtensible Markup Language (XML)* data comprising:	Challenger does not teach this claim element.
a data store for storing *XML* data;	Challenger does not teach this claim element.
a server for retrieving and updating *XML* data in the data store to service client requests;	Challenger does not teach this claim element.
a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules;	Challenger does not teach this claim element.
a cache for temporarily storing transformed *XML* data as data objects for later reuse;	Challenger does not teach this claim element.
a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server; and	Challenger does not teach this claim element.
an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules.	Challenger does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Challenger has been generalized, and, in fact, does not support the position asserted by the Examiner.

a data store for storing *XML* data

In particular, Challenger fails to teach "a data store for storing *XML* data", as required by claim 1. Instead, Challenger discloses a database such that "underlying data [is] stored on a database 99." (See Challenger, column 8, lines 41-42 and Figure 1A.) Thus, Challenger fails to teach that database 99 can store *XML* data. Therefore, Challenger does not teach the claim 1 element of "a data store for storing *XML* data".

a server for retrieving and updating *XML* data in the data store to service client requests

In particular, Challenger fails to teach "a server for retrieving and updating *XML* data in the data store to service client requests", as required by claim 1. Instead, Challenger discloses a server such that "a client 90 communicates requests to a server 100 over a network 95." (See Challenger, column 8, lines 24-25 and Figure 1A.) Thus, Challenger fails to teach that server 100 can retrieve and update *XML* data. Therefore, Challenger does not teach the claim 1 element of "a server for retrieving and updating *XML* data in the data store to service client requests".

a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules

In particular, Challenger fails to teach "a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules", as required by claim 1. Instead, Challenger discloses "API's (FIG. 4) which allow an application 97 program to specify the records that a cached object depends upon." (See Challenger, column 10, lines 15-17.) Thus, Challenger fails to teach that the API's can transform "*XML* data into a format suitable for a client application based on a set of transformation rules." Therefore, Challenger does not teach the claim 1 element of "a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules".

a cache for temporarily storing transformed *XML* data as data objects for later reuse

In particular, Challenger fails to teach "a cache for temporarily storing transformed *XML* data as data objects for later reuse", as required by claim 1. Instead, Challenger discloses (1) "Local Cache[,which] . . . is a cache (or other standard object store such as a file system) which is updated by an instance of a Trigger Monitor residing on the same physical machine as the cache itself[, and] Remote Cache[, which] . . . is a cache (or other standard object store such as a file system) which is updated by an instance of a Trigger Monitor residing on a different physical machine from the cache itself." (See Challenger, column 29, lines 52-61.) Thus, Challenger fails to teach that the either the Local Cache or the Remote Cache can be used "for temporarily storing transformed *XML* data as data objects for later reuse". Therefore, Challenger does not teach the claim 1 element of "a cache for temporarily storing transformed *XML* data as data objects for later reuse".

a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server

In particular, Challenger fails to teach "a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server", as required by claim 1. Instead, Challenger discloses "a cache manager 1 (which is an example of an object manager) determines how changes to underlying data affect the values of objects." (See Challenger, column 8, lines 54-56.) Thus, Challenger fails to teach that cache manager 1 can be used "for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server". Therefore, Challenger does not teach the claim 1 element of "a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server".

an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules

In particular, Challenger fails to teach "an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules", as required by claim 1. Instead, Challenger discloses an "object manager [that] maintains the data dependence information . . . [such that] [w]henever new objects are created or the data dependencies change, the object

manager is responsible for updating the appropriate information." (See Challenger, column 4, lines 33-37.) Thus, Challenger fails to teach that the object manager can be used "for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules". Therefore, Challenger does not teach the claim 1 element of "an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules". It is therefore clear that Challenger cannot teach each element of claim 1 and, therefore, a rejection of claim 1 under 35 U.S.C. § 102(b) is inappropriate.

Claim 2

Since dependent claim 2 depends on independent claim 1 and since Challenger cannot teach each element of claim 1, Challenger also cannot teach each element of claim 2, and, therefore, a rejection of claim 2 under 35 U.S.C. § 102(b) is inappropriate.

Claim 3

Since dependent claim 3 depends on dependent claim 2 and since Challenger cannot teach each element of claim 2, Challenger also cannot teach each element of claim 3, and therefore, a rejection of claim 3 under 35 U.S.C. § 102(b) is inappropriate.

Claim 4

Since dependent claim 4 depends on dependent claim 3 and since Challenger cannot teach each element of claim 3, Challenger also cannot teach each element of claim 4, and, therefore, a rejection of claim 4 under 35 U.S.C. § 102(b) is inappropriate.

Claim 5

Since dependent claim 5 depends on dependent claim 4 and since Challenger cannot teach each element of claim 4, Challenger also cannot teach each element of claim 5, and therefore, a rejection of claim 5 under 35 U.S.C. § 102(b) is inappropriate.

Claim 6

Since dependent claim 6 depends on dependent claim 5 and since Challenger cannot teach each element of claim 5, Challenger also cannot teach each element of claim 6, and, therefore, a rejection of claim 6 under 35 U.S.C. § 102(b) is inappropriate.

Claim 7

Since dependent claim 7 depends on dependent claim 6 and since Challenger cannot teach each element of claim 6, Challenger also cannot teach each element of claim 7, and therefore, a rejection of claim 7 under 35 U.S.C. § 102(b) is inappropriate.

Claims 8 and 12

Since dependent claims 8 and 12 depend on dependent claim 7 and since Challenger cannot teach each element of claim 7, Challenger also cannot teach each element of claim 8 or 12, and therefore, a rejection of claim 8 or 12 under 35 U.S.C. § 102(b) is inappropriate.

Claim 9

Since dependent claim 9 depends on dependent claim 8 and since Challenger cannot teach each element of claim 8, Challenger also cannot teach each element of claim 9, and therefore, a rejection of claim 9 under 35 U.S.C. § 102(b) is inappropriate.

Claims 10 and 11

Since dependent claims 10 and 11 depend on dependent claim 9 and since Challenger cannot teach each element of claim 9, Challenger also cannot teach each element of claim 10 or 11, and therefore, a rejection of claim 10 or 11 under 35 U.S.C. § 102(b) is inappropriate.

Claims 13, 14, and 15

Since dependent claims 13, 14, and 15 depend on dependent claim 12 and since Challenger cannot teach each element of claim 12, Challenger also cannot teach each element of claim 13, 14, or 15, and therefore, a rejection of claim 13, 14, or 15 under 35 U.S.C. § 102(b) is inappropriate.

Claims 16-21

Claim 16

The Examiner has asserted that

> Challenger discloses in figures 1, 9, 12, and 30, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of

transformation rules; a cache for temporarily storing transformed data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (Abstract; col.4, lines 6-10; col. 10, lines 14-24; col. 29, lines 32 +).

(See Office Action, page 2.)

To the extent the Examiner's language at page 2 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Challenger and claim 16:

Claim 16	**Challenger**
16. In a client-sever computing system having a cache and storing *eXtensible Markup Language (XML)* data as data objects, a method for determining invalid cached objects comprising:	Challenger does not teach this claim element.
transforming *XML* data into a format suitable for a client application based on a set of transformation rules;	Challenger does not teach this claim element.
determining dependencies between cached objects and *XML* data related to the cached objects;	Challenger does not teach this claim element.
monitoring updates to the related *XML* data; and	Challenger does not teach this claim element.
determining the cached objects that are affected by changes to the related *XML* data based on the dependencies.	Challenger does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Challenger has been generalized, and, in fact, does not support the position asserted by the Examiner.

transforming *XML* data into a format suitable for a client application based on a set of transformation rules

In particular, Challenger fails to teach "transforming *XML* data into a format suitable for a client application based on a set of transformation rules", as required by claim 16. Instead, Challenger discloses "API's (FIG. 4) which allow an application 97 program to specify the records that a cached object depends upon." (See Challenger, column 10, lines 15-17.) Thus, Challenger fails to teach that the API's can be used for "transforming *XML* data into a format suitable for a client application based on a set of transformation rules." Therefore, Challenger does not teach the claim 16 element of "transforming *XML* data into a format suitable for a client application based on a set of transformation rules".

determining dependencies between cached objects and *XML* data related to the cached objects

In particular, Challenger fails to teach "determining dependencies between cached objects and *XML* data related to the cached objects", as required by claim 16. Instead, Challenger discloses an "object manager [that] maintains the data dependence information . . . [such that] [w]henever new objects are created or the data dependencies change, the object manager is responsible for updating the appropriate information." (See Challenger, column 4, lines 33-37.) Thus, Challenger fails to teach that the object manager can be used for "determining dependencies between cached objects and *XML* data related to the cached objects". Therefore, Challenger does not teach the claim 16 element of "determining dependencies between cached objects and *XML* data related to the cached objects".

monitoring updates to the related *XML* data

In particular, Challenger fails to teach "monitoring updates to the related *XML* data", as required by claim 16. Instead, Challenger discloses "a cache manager 1 (which is an example of an object manager) determines how changes to underlying data affect the values of objects." (See Challenger, column 8, lines 54-56.) Thus, Challenger fails to

teach that cache manager 1 can be used for "monitoring updates to the related *XML* data". Therefore, Challenger does not teach the claim 16 element of "monitoring updates to the related *XML* data".

determining the cached objects that are affected by changes to the related *XML* data based on the dependencies

In particular, Challenger fails to teach "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies", as required by claim 16. Instead, Challenger discloses an "object manager [that] maintains the data dependence information . . . [such that] [w]henever new objects are created or the data dependencies change, the object manager is responsible for updating the appropriate information." (See Challenger, column 4, lines 33-37.) Thus, Challenger fails to teach that the object manager can be used for "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". Therefore, Challenger does not teach the claim 16 element of "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". It is therefore clear that Challenger cannot teach each element of claim 16 and, therefore, a rejection of claim 16 under 35 U.S.C. § 102(b) is inappropriate.

Claims 17, 18, and 19

Since dependent claims 17, 18, and 19 depend on independent claim 16 and since Challenger cannot teach each element of claim 16, Challenger also cannot teach each element of claim 17, 18, or 19 and therefore, a rejection of claim 17, 18, or 19 under 35 U.S.C. § 102(b) is inappropriate.

Claim 20

Since dependent claim 20 depends on dependent claim 19 and since Challenger cannot teach each element of claim 19, Challenger also cannot teach each element of claim 20, and therefore, a rejection of claim 20 under 35 U.S.C. § 102(b) is inappropriate.

Claim 21

Since dependent claim 21 depends on dependent claim 20 and since Challenger cannot teach each element of claim 20, Challenger also cannot teach each element of claim 21, and therefore, a rejection of claim 21 under 35 U.S.C. § 102(b) is inappropriate.

Claims 22-27

Claim 22

Since claim 22, as amended, is the computer program product version of claim 16 with similar elements as claim 16, since the Examiner has asserted arguments against Claim 16 that are similar to the arguments asserted by the Examiner against claim 16, and since Challenger cannot teach each element of claim 16, Challenger also cannot teach each element of claim 22 for similar reasons that Challenger cannot teach each element of claim 16, and therefore, a rejection of claim 22 under 35 U.S.C. § 102(b) is inappropriate.

Claims 23, 24, and 25

Since dependent claims 23, 24, and 25 depend on independent claim 22 and since Challenger cannot teach each element of claim 2, Challenger also cannot teach each element of claim 23, 24, or 25 and therefore, a rejection of claim 23, 24, or 25 under 35 U.S.C. § 102(b) is inappropriate.

Claim 26

Since dependent claim 26 depends on dependent claim 25 and since Challenger cannot teach each element of claim 25, Challenger also cannot teach each element of claim 26, and therefore, a rejection of claim 26 under 35 U.S.C. § 102(b) is inappropriate.

Claim 27

Since dependent claim 27 depends on dependent claim 26 and since Challenger cannot teach each element of claim 26, Challenger also cannot teach each element of claim 27, and therefore, a rejection of claim 27 under 35 U.S.C. § 102(b) is inappropriate.

Kelley

By the Office Action dated August 23, 2006, the Examiner has rejected claims 1, 16, and 22 under 35 U.S.C. § 102(b) as being anticipated by Kelley (U.S. Patent No. 6,088,659) (hereinafter "Kelley"). In order to be an anticipation of a claim under 35 U.S.C. § 102(b), a reference must teach every element of the claim, including the relationships between the elements. If any element is not fully taught by the reference, the rejection cannot be sustained.

Evaluating Kelley in this light, it is appropriate to examine the portions of Kelley which the Examiner has pointed to as teaching the claimed elements.

Claim 1

The Examiner has asserted that

> Kelley discloses in figures 3-8, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of transformation rules; a cache for temporarily storing transformed data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (col. 7, lines 52+; col. 14, lines 26-61; col. 15, lines 26-65; col. 16, lines 60+; col. 34, lines 30-46).

(See Office Action, page 5.)

To the extent the Examiner's language at page 5 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Kelley and claim 1:

Claim 1	**Kelley**
1. A client-server system capable of validating cached *eXtensible Markup Language (XML)* data comprising:	Kelley does not teach this claim element.
a data store for storing *XML* data;	Kelley does not teach this claim element.
a server for retrieving and updating *XML* data in the data store to service client requests;	Kelley does not teach this claim element.
a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules;	Kelley does not teach this claim element.
a cache for temporarily storing	Kelley does not teach this claim element.

transformed *XML* data as data objects for later reuse;	
a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server; and	Kelley does not teach this claim element.
an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules.	Kelley does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Kelley has been generalized, and, in fact, does not support the position asserted by the Examiner.

a data store for storing *XML* data

In particular, Kelley fails to teach "a data store for storing *XML* data", as required by claim 1. Instead, Kelley discloses a database such that "encapsulated business objects . . . are persistently stored in the database." (See Kelley, column 13, lines 61-62.) Thus, Kelley fails to teach that the database can store *XML* data. Therefore, Kelley does not teach the claim 1 element of "a data store for storing *XML* data".

a server for retrieving and updating *XML* data in the data store to service client requests

In particular, Kelley fails to teach "a server for retrieving and updating *XML* data in the data store to service client requests", as required by claim 1. Instead, Kelley discloses "an AMR Server 15 that collects, loads, and manages system-wide metering data from electronic or electromechanical meters 60 located at customers' premisses 70 and routes it automatically to upstream business systems 50." (See Kelley, column 12, lines 14-18.) Thus, Kelley fails to teach that Server 15 can retrieve and update *XML* data. Therefore, Kelley does not teach the claim 1 element of "a server for retrieving and updating *XML* data in the data store to service client requests".

a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules

In particular, Kelley fails to teach "a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules", as required by claim 1. Instead, Kelley discloses "a Mapping Subsystem 140 [that] provides services that allow easy customization of file formats for exporting data from and importing data to the AMR Server 15." (See Kelley, column 31, lines 13-16.) Thus, Kelley fails to teach that Mapping Subsystem 140 can transform "*XML* data into a format suitable for a client application based on a set of transformation rules." Therefore, Kelley does not teach the claim 1 element of "a transformation engine for transforming *XML* data into a format suitable for a client application based on a set of transformation rules".

a cache for temporarily storing transformed *XML* data as data objects for later reuse

In particular, Kelley fails to teach "a cache for temporarily storing transformed *XML* data as data objects for later reuse", as required by claim 1. Instead, Kelley discloses a cache such that "DAOs reside in cache for fast access." (See Kelley, column 44, lines 23-24.) Thus, Kelley fails to teach that the cache can be used "for temporarily storing transformed *XML* data as data objects for later reuse". Therefore, Kelley does not teach the claim 1 element of "a cache for temporarily storing transformed *XML* data as data objects for later reuse".

an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules

In particular, Kelley fails to teach "an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules", as required by claim 1. Instead, Kelley discloses that "Database access dependencies are preferably handled by ScheduleView and kept transparent to ScheduleBuilder." (See Kelley, column 28, lines 40-42.) Thus, Kelley fails to teach that ScheduleView can be used "for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules". Therefore, Kelley does not teach the claim 1 element of "an object dependency mapper for automatically and

continuously determining dependencies between *XML* data in the data store and sets of transformation rules". It is therefore clear that Kelley cannot teach each element of claim 1 and, therefore, a rejection of claim 1 under 35 U.S.C. § 102(b) is inappropriate.

Claim 16

The Examiner has asserted that

> Kelley discloses in figures 3-8, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of transformation rules; a cache for temporarily storing transformed data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (col. 7, lines 52+; col. 14, lines 26-61; col. 15, lines 26-65; col. 16, lines 60+; col. 34, lines 30-46).

(See Office Action, page 5.)

To the extent the Examiner's language at page 5 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Kelley and claim 16:

Claim 16	**Kelley**
16. In a client-sever computing system having a cache and storing *eXtensible Markup Language (XML)* data as data objects, a method for determining invalid cached objects comprising:	Kelley does not teach this claim element.
transforming *XML* data into a format suitable for a client application	Kelley does not teach this claim element.

based on a set of transformation rules;	
determining dependencies between cached objects and *XML* data related to the cached objects;	Kelley does not teach this claim element.
monitoring updates to the related *XML* data; and	Kelley does not teach this claim element.
determining the cached objects that are affected by changes to the related *XML* data based on the dependencies.	Kelley does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Kelley has been generalized, and, in fact, does not support the position asserted by the Examiner.

transforming *XML* data into a format suitable for a client application based on a set of transformation rules

In particular, Kelley fails to teach "transforming *XML* data into a format suitable for a client application based on a set of transformation rules", as required by claim 16. Instead, Kelley discloses "a Mapping Subsystem 140 [that] provides services that allow easy customization of file formats for exporting data from and importing data to the AMR Server 15." (See Kelley, column 31, lines 13-16.) Thus, Kelley fails to teach that Mapping Subsystem 140 can transform "*XML* data into a format suitable for a client application based on a set of transformation rules." Therefore, Kelley does not teach the claim 16 element of "transforming *XML* data into a format suitable for a client application based on a set of transformation rules".

determining dependencies between cached objects and *XML* data related to the cached objects

In particular, Kelley fails to teach "determining dependencies between cached objects and *XML* data related to the cached objects", as required by claim 16. Instead, Kelley discloses that "Database access dependencies are preferably handled by ScheduleView and kept transparent to ScheduleBuilder." (See Kelley, column 28, lines 40-42.) Thus, Kelley fails to teach that ScheduleView can be used for "determining

dependencies between cached objects and *XML* data related to the cached objects". Therefore, Kelley does not teach the claim 16 element of "determining dependencies between cached objects and *XML* data related to the cached objects".

determining the cached objects that are affected by changes to the related *XML* data based on the dependencies

In particular, Kelley fails to teach "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies", as required by claim 16. Instead, Kelley discloses that "Database access dependencies are preferably handled by ScheduleView and kept transparent to ScheduleBuilder." (See Kelley, column 28, lines 40-42.) Thus, Kelley fails to teach that ScheduleView can be used for "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". Therefore, Kelley does not teach the claim 16 element of "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". It is therefore clear that Kelley cannot teach each element of claim 16 and, therefore, a rejection of claim 16 under 35 U.S.C. § 102(b) is inappropriate.

Claim 22

Since claim 22, as amended, is the computer program product version of claim 16 with similar elements as claim 16, since the Examiner has asserted arguments against Claim 16 that are similar to the arguments asserted by the Examiner against claim 16, and since Kelley cannot teach each element of claim 16, Kelley also cannot teach each element of claim 22 for similar reasons that Kelley cannot teach each element of claim 16, and therefore, a rejection of claim 22 under 35 U.S.C. § 102(b) is inappropriate.

Reed

By the Office Action dated August 23, 2006, the Examiner has rejected claims 1, 16, and 22 under 35 U.S.C. § 102(b) as being anticipated by Reed (U.S. Patent Application Publication No. US 2002/01289966 A1) (hereinafter "Reed"). In order to be an anticipation of a claim under 35 U.S.C. § 102(e), a reference must teach every element of the claim, including the relationships between the elements. If any element is not fully taught by the reference, the rejection cannot be sustained.

Evaluating Reed in this light, it is appropriate to examine the portions of Reed which the Examiner has pointed to as teaching the claimed elements.

Claim 1

The Examiner has asserted that

> Reed discloses in figures 1, 3, 6, and 8, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of transformation rules; a cache for temporarily storing transformed data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (Paragraph [0013], [0046], and in claims 2, 8, and 10.

(See Office Action, page 6.)

To the extent the Examiner's language at page 5 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Reed and claim 1:

Claim 1	**Reed**
1. A client-server system capable of validating cached *eXtensible Markup Language (XML)* data comprising:	Reed does not teach this claim element.
a data store for storing *XML* data;	Reed does not teach this claim element.
a server for retrieving and updating *XML* data in the data store to service client requests;	Reed does not teach this claim element.
a transformation engine for transforming *XML* data into a format suitable for a client application based on a	Reed does not teach this claim element.

set of transformation rules; a cache for temporarily storing transformed *XML* data as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between *XML* data in the data store and sets of transformation rules.	Reed does not teach this claim element. Reed does not teach this claim element. Reed does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Reed has been generalized, and, in fact, does not support the position asserted by the Examiner.

a data store for storing *XML* data

In particular, Reed fails to teach "a data store for storing *XML* data", as required by claim 1. Instead, Reed discloses "[a]n operational data store (ODS) 11 [that] stores production data generated by a production system 12." (See Reed, paragraph [0026].) Thus, Reed fails to teach that data store 11 can store *XML* data. Therefore, Reed does not teach the claim 1 element of "a data store for storing *XML* data".

a server for retrieving and updating *XML* data in the data store to service client requests

In particular, Reed fails to teach "a server for retrieving and updating *XML* data in the data store to service client requests", as required by claim 1. Instead, Reed discloses "a dedicated transaction server 14 that provides a high throughput interface to the operational data store 11." (See Reed, paragraph 27.) Thus, Reed fails to teach that server 14 can retrieve and update *XML* data. Therefore, Reed does not teach the claim 1 element of "a server for retrieving and updating *XML* data in the data store to service client requests".

a cache for temporarily storing transformed *XML* data as data objects for later reuse

In particular, Reed fails to teach "a cache for temporarily storing transformed *XML* data as data objects for later reuse", as required by claim 1. Instead, Reed discloses "a cache 60 for staging information, including log entries and updated records 57." (See Reed, paragraph [0046].) Thus, Reed fails to teach that cache 60 can be used "for temporarily storing transformed *XML* data as data objects for later reuse". Therefore, Reed does not teach the claim 1 element of "a cache for temporarily storing transformed *XML* data as data objects for later reuse".

a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server

In particular, Reed fails to teach "a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server", as required by claim 1. Instead, Reed discloses "a log monitor 26 [that] can provide large-grained database concurrency between the operational data store 11 and enterprise data warehouse 16 by indirectly updating the informational data." (See Reed, paragraph [0032].) Thus, Reed fails to teach that log monitor 26 can be used "for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server". Therefore, Reed does not teach the claim 1 element of "a cache monitor for ensuring that cached objects are validated when changes to *XML* data in the data store are detected by the server". It is therefore clear that Reed cannot teach each element of claim 1 and, therefore, a rejection of claim 1 under 35 U.S.C. § 102(e) is inappropriate.

Claim 16

The Examiner has asserted that

> Reed discloses in figures 1, 3, 6, and 8, a client-server system capable of validating cached data comprising a data store for storing data; a server for retrieving and updating data in the data store to service client requests; a transformation engine for transforming data into a format suitable for a client application based on a set of transformation rules; a cache for temporarily storing transformed data

as data objects for later reuse; a cache monitor for ensuring that cached objects are validated when changes to data in the data store are detected by the server; and an object dependency mapper for automatically and continuously determining dependencies between data in the data store and sets of transformation rules (Paragraph [0013], [0046], and in claims 2, 8, and 10.

(See Office Action, page 6.)

To the extent the Examiner's language at page 5 of the Office Action can be understood, it appears that the Examiner has asserted the following correspondence between Reed and claim 16:

Claim 16	**Reed**
16. In a client-sever computing system having a cache and storing *eXtensible Markup Language (XML)* data as data objects, a method for determining invalid cached objects comprising:	Reed does not teach this claim element.
transforming *XML* data into a format suitable for a client application based on a set of transformation rules;	Reed does not teach this claim element.
determining dependencies between cached objects and *XML* data related to the cached objects;	Reed does not teach this claim element.
monitoring updates to the related *XML* data; and	Reed does not teach this claim element.
determining the cached objects that are affected by changes to the related *XML* data based on the dependencies.	Reed does not teach this claim element.

In reviewing the cited portions of original, however, it becomes apparent that Reed has been generalized, and, in fact, does not support the position asserted by the Examiner.

determining dependencies between cached objects and *XML* data related to the cached objects

In particular, Reed fails to teach "determining dependencies between cached objects and *XML* data related to the cached objects", as required by claim 16. Instead, Reed discloses "a log monitor 26 [that] can provide large-grained database concurrency between the operational data store 11 and enterprise data warehouse 16 by indirectly updating the informational data." (See Reed, paragraph [0032].) Thus, Reed fails to teach that log monitor 26 can be used for "determining dependencies between cached objects and *XML* data related to the cached objects". Therefore, Reed does not teach the claim 16 element of "determining dependencies between cached objects and *XML* data related to the cached objects".

determining the cached objects that are affected by changes to the related *XML* data based on the dependencies

In particular, Reed fails to teach "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies", as required by claim 16. Instead, Reed discloses "a log monitor 26 [that] can provide large-grained database concurrency between the operational data store 11 and enterprise data warehouse 16 by indirectly updating the informational data." (See Reed, paragraph [0032].) Thus, Reed fails to teach that log monitor 26 can be used for "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". Therefore, Reed does not teach the claim 16 element of "determining the cached objects that are affected by changes to the related *XML* data based on the dependencies". It is therefore clear that Reed cannot teach each element of claim 16 and, therefore, a rejection of claim 16 under 35 U.S.C. § 102(e) is inappropriate.

Claim 22

Since claim 22, as amended, is the computer program product version of claim 16 with similar elements as claim 16, since the Examiner has asserted arguments against Claim 16 that are similar to the arguments asserted by the Examiner against claim 16, and since Reed cannot teach each element of claim 16, Reed also cannot teach each element of claim 22 for similar reasons that Reed cannot teach each element of claim 16, and therefore, a rejection of claim 22 under 35 U.S.C. § 102(e) is inappropriate.

Conclusion

It is therefore clear that claims 1-27 comply with the requirements of 35 U.S.C. §§ 101, 102, 103, and 112. The application is therefore in condition for allowance. Early notification to that effect is respectfully solicited.

In the event that any issue remains unresolved, the Examiner is invited to telephone the undersigned at 408-927-3377.

Respectfully Submitted,

Date: November 24, 2006

Leonard T. Guzman
Reg. No. 46,308

IBM Almaden Research Center
650 Harry Road
C45A/J2B
San Jose, CA 95120

Phone Number: 408-927-3377
Facsimile Number: 408-927-3375